

(19) World Intellectual Property
Organization
International Bureau



(43) International Publication Date
25 November 2004 (25.11.2004)

PCT

(10) International Publication Number
WO 2004/102403 A2

(51) International Patent Classification⁷: **G06F 13/16**,
13/42

68 Stewart Street, Franklin, MA 02038 (US). **MIRANDA, Paul, C.** [US/US]; 4836 Canyonbend Circle, Austin, TX • 78735 (US).

(21) International Application Number:
PCT/US2004/014441

(74) Agent: **DRAKE, Paul, S.**; Advanced Micro Devices, Inc., 5204 East Ben White Boulevard, Mail Stop 562, Austin, TX 78741 (US).

(22) International Filing Date: 10 May 2004 (10.05.2004)

(25) Filing Language: English

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(26) Publication Language: English

(30) Priority Data:
60/470,078 13 May 2003 (13.05.2003) US

(84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

(71) Applicant (for all designated States except US): **ADVANCED MICRO DEVICES, INC.** [US/US]; One AMD Place, Mail Stop 68, P.O. Box 3453, Sunnyvale, CA 94088-3453 (US).

(72) Inventors; and

(75) Inventors/Applicants (for US only): **POLZIN, R., Stephen** [US/US]; 345 Brookwood Avenue, San Jose, CA 95116 (US). **WEBER, Frederick, D.** [US/US]; 1137 Settle Avenue, San Jose, CA 95125 (US). **TALBOT, Gerald, R.** [GB/US]; 280 Holdenwood Road, Concord, MA 01742 (US). **HEWITT, Larry, D.** [US/US]; 6103 Bend O River Drive, Austin, TX 78746 (US). **REEVES, Richard, W.** [US/US]; 9 Belknap Street, Westboro, MA 01581 (US). **PATEL, Shwetal, A.** [US/US]; 3677 Jasmine Circle, San Jose, CA 95135 (US). **LA FETRA, Ross, V.** [US/US]; 1562 Oriole Avenue, Sunnyvale, CA 94087 (US). **GULICK, Dale, E.** [US/US]; 11715 Astoria, Austin, TX 78738 (US). **HUMMEL, Mark, D.** [US/US];

Published:

— without international search report and to be republished upon receipt of that report

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: A SYSTEM INCLUDING A HOST CONNECTED TO A PLURALITY OF MEMORY MODULES VIA A SERIAL MEMORY INTERCONNECT

(57) Abstract: A system (50) including a host (100) coupled to a serially connected chain of memory modules (150A-B). In one embodiment, each of the memory modules includes a memory control hub (160) for controlling access to a plurality of memory chips (261) on the memory module. The memory modules are coupled serially in a chain to the host via a plurality of memory links (110). Each memory link may include an uplink (211) for conveying transactions toward the host and a downlink (212) for conveying transactions originating at the host to a next memory module in the chain. The uplink and the downlink may convey transactions using packets that include control and configuration packets and memory access packets. The memory control hub may convey a transaction received on a first downlink of a first memory link on a second downlink of a second memory link independent of decoding the transaction.

WO 2004/102403 A2

**TITLE: A SYSTEM INCLUDING A HOST CONNECTED TO A PLURALITY OF MEMORY
MODULES VIA A SERIAL MEMORY INTERCONNECT**

Technical Field

- 5 **[0001]** This invention relates to computer system memory and, more particularly, to memory module configurations and the memory subsystem topology.

Background Art

- 10 **[0002]** Many computer systems employ a main system memory that may be configured dependent upon the needs of an end user. In such systems, a motherboard or system board may include a number of memory expansion sockets. One or more small circuit boards, referred to as memory modules, may be inserted into the sockets as needed to increase the memory capacity of the computer system. Each of the memory modules typically includes multiple memory devices that provide a given amount of memory capacity. The memory devices are usually implemented using some type of dynamic random access memory (DRAM). Some examples of DRAM types
15 include synchronous DRAM (SDRAM) as well as the various types of double data rate SDRAM (DDR SDRAM).

- [0003]** In conventional computer systems, the memory modules are connected to a memory/DRAM controller via a memory bus that includes address, control and a data signals. In some computer systems, the address, control and data signals may be multiplexed and thus share the same sets of wires. In other computer systems, the address, control and data signals may use separate wires. In either case, each of the address and control signals are routed to
20 each expansion socket such that the memory modules, when inserted, are connected in parallel to the memory/DRAM controller. In some systems the memory/DRAM controller may reside on the same integrated circuit (IC) chip as the system processor, while in other systems the memory/DRAM controller may reside in one IC (e.g., a Northbridge) of a chipset.

- [0004]** Although the operating speed of computer system processors continues to increase, the relative
25 performance of the main system memory has not increased at the same rate. This may be due, at least in part, to the incremental improvement in the bandwidth of the memory bus architectures described above.

DISCLOSURE OF INVENTION

- [0005]** Various embodiments of a system including a plurality of memory modules are disclosed. In one
30 embodiment, a host is coupled to a serially connected chain of memory modules.

- [0006]** In one specific implementation, each of the memory modules may include a memory control hub that may control access to a plurality of memory chips on the memory module. In addition, the memory modules are coupled serially in a chain to the host via a plurality of memory links. Each memory link may include an uplink for conveying transactions toward the host and a downlink for conveying transactions originating at the host to a next
35 memory module in the chain. The uplink and the downlink may each be a uni-directional link including a plurality of signals that may convey transactions using packets that include control and configuration packets and memory access packets.

[0007] In another specific implementation, the memory control hub may decode a transaction received on a first downlink of a first memory link. In addition, the memory control hub may convey the transaction on a second downlink of a second memory link independent of decoding the transaction.

[0008] In still another specific implementation, the memory control hub may include a DRAM controller that may access the memory chips in response to decoding the transaction into a memory command having a memory address that matches a memory address associated with the memory control hub.

BRIEF DESCRIPTION OF DRAWINGS

[0009] FIG. 1 is a block diagram of one embodiment of a system including a serially connected chain of memory modules.

[0010] FIG. 2 is a block diagram of one embodiment of a memory module such as a memory module illustrated in FIG. 1.

[0011] FIG. 3 is a block diagram of one embodiment of a downlink control unit such as the downlink control unit illustrated in FIG. 2.

[0012] FIG. 4 is a block diagram of one embodiment of an uplink control unit such as the uplink control unit illustrated in FIG. 2.

[0013] FIG. 5A is a diagram of one embodiment of a configuration read packet.

[0014] FIG. 5B is a diagram of one embodiment of a configuration write packet.

[0015] FIG. 5C is a diagram of one embodiment of a memory read packet.

[0016] FIG. 5D is a diagram of one embodiment of a memory write packet.

[0017] FIG. 6 is a block diagram of one embodiment of a computer system.

[0018] While the invention is susceptible to various modifications and alternative forms, specific embodiments thereof are shown by way of example in the drawings and will herein be described in detail. It should be understood, however, that the drawings and detailed description thereto are not intended to limit the invention to the particular form disclosed, but on the contrary, the intention is to cover all modifications, equivalents, and alternatives falling within the spirit and scope of the present invention as defined by the appended claims. Note, the headings are for organizational purposes only and are not meant to be used to limit or interpret the description or claims. Furthermore, note that the word "may" is used throughout this application in a permissive sense (i.e., having the potential to, being able to), not a mandatory sense (i.e., must). The term "include" and derivations thereof mean "including, but not limited to." The term "connected" means "directly or indirectly connected," and the term "coupled" means "directly or indirectly coupled."

MODE(S) FOR CARRYING OUT THE INVENTION

[0019] Turning now to FIG. 1, a block diagram of one embodiment of a system including a serially connected chain of memory modules is shown. System 50 includes a host 100 coupled to a system memory 125 via a memory link 110A. System 50 may be configured to operate as part of a computing device such as a computer system or server system, for example. System memory 125 includes a memory module 150A coupled to a memory module 150B via a memory link 110B. Memory module 150B is shown coupled to a memory link 110C, which may be

coupled to an additional memory module (not shown) as desired to form a serially connected chain of memory modules that is coupled to host 100. It is noted that although two memory modules are shown in the chain, it is contemplated that one or more memory modules may be connected in this manner. It is further noted that components including a reference number followed by a reference letter may be referred to generally by the reference number alone. For example, when referring generally to all memory modules, reference may be made to memory module 150.

[0020] In the illustrated embodiment, memory module 150A includes a memory control hub 160A, which is coupled to a plurality of memory devices that are designated memory chip 171A through 171N, where N may be any number, as desired. In one embodiment, memory control hub 160A may be coupled to the memory chips via any type of memory interconnect. For example, in one embodiment, the memory interconnect may be a typical address, control and data bus configuration.

[0021] Similarly, memory module 150B includes a memory control hub 160B, which is coupled to a plurality of memory devices that are designated memory chip 181A through 181N, where N may be any number, as desired. In one embodiment, memory control hub 160B may be coupled to the memory chips via any type of memory interconnect as described above. It is noted that each of memory chips 171A through 171N and 181A through 181N may be any type of memory device such as a memory device in the DRAM family of memory devices, for example.

[0022] In the illustrated embodiment, memory links 110A-110C form a memory interconnect. In one embodiment, each of memory links 110A-110C forms a point-to-point memory interconnect that is implemented as two sets of unidirectional lines. One set of unidirectional lines is referred to as a downlink and is configured to convey transactions away from host 100 in a downstream direction. The other set of unidirectional lines is referred to as an uplink and is configured to convey transactions toward host 100 in an upstream direction. In addition, in one embodiment, each set of unidirectional lines may be implemented using a plurality of differential signal pairs. In one embodiment, each memory link 110 includes an 18-bit downlink and a 16-bit uplink, where each bit is a differential signal pair. As will be described in greater detail below in conjunction with the descriptions of FIG. 5A through FIG. 5D, the memory interconnect formed by memory links 110 may be configured to convey packets.

[0023] Generally speaking, all transactions from host 100 flow downstream through all memory modules 150 on the downlink and all response transactions flow upstream from the responding memory module 150 through each upstream memory module 150 on the uplink. More particularly, in one embodiment, host 100 may request to retrieve or store data within system memory 125. In response to host 100 making a request, memory controller 105 initiates a corresponding transaction such as a memory read transaction or a memory write transaction, for example. Memory controller 105 transmits the transaction to system memory 125 via memory link 110A. In the illustrated embodiment, the transaction is received by memory control hub 160A of memory module 150A.

[0024] In response to receiving the transaction, memory control hub 160A is configured to transmit the received transaction to memory module 150B via memory link 110B without decoding the transaction. This is referred to as forwarding the transaction downstream. Thus, each transaction received on a downlink by a given memory control hub 160 of a given memory module 150 is forwarded to the next memory module 150 in the chain that is coupled to the downlink without decoding the transaction. In one embodiment, decoding of the transaction may occur in parallel with the forwarding of the transaction. In other embodiments, the decoding of the transaction

may occur after the transaction has been forwarded. A more detailed description of downstream forwarding function may be found below in the description of FIG. 3.

[0025] Likewise, if memory controller 105 initiates a read request transaction, for example, the memory module 150 having the memory location corresponding to the address in the request will respond with the requested data. The response will be transmitted on the memory module's uplink toward host 100. If there are any intervening memory modules between the sending memory module and host 100, the intervening memory module will forward the response transaction on its uplink to either host 100 or the next memory module in the chain in an upstream direction. In addition, when the responding memory module is ready to send the response, it may inject the response into a sequence of transactions that are being forwarded upstream on the uplink. A more detailed description of upstream forwarding function may be found below in the description of FIG. 5.

[0026] In one embodiment, memory controller 105 may be configured to make requests to system memory 125 without knowledge of which of memory modules 150A and 150B a particular address is associated. For example, each of memory modules 150 may be assigned a range of memory addresses during a system configuration sequence. Each memory control hub 160 may include logic (not shown in FIG. 1) that may decode the address of an incoming request. Thus, a memory control hub 160 of a given memory module 150 may initiate a memory read cycle or memory write cycle to the memory chips on the given memory module 150 in response to decoding a memory request having an address that is in the address range assigned to the given memory module 150. As will be described in greater detail below in conjunction with the description of FIG. 2, in one embodiment, each memory control hub 160 may include a DRAM controller (not shown in FIG. 1) for initiating memory cycles to the memory chips to which it is connected.

[0027] In addition, in one embodiment, memory controller 105 may initiate a subsequent memory access request prior to receiving a response to a previous memory access request. In such an embodiment, memory controller 105 may keep track of outstanding requests and may thus process the responses in a different order than they were sent.

[0028] It is noted that in an alternative embodiment, each of memory links 110 may form a point-to-point memory interconnect that is implemented as one set of bi-directional lines. As such, transactions may flow both upstream and downstream on the set of bi-directional wires. In such an embodiment, the bi-directional lines may be implemented using a plurality of differential signal pairs.

[0029] Referring to FIG. 2, a block diagram of one embodiment of a memory module such as the memory module illustrated in FIG. 1 is shown. Components that correspond to those shown in FIG. 1 are numbered identically for clarity and simplicity. Memory module 150 includes a memory control hub 160 coupled to memory chips 261A through 261N via a memory bus 265. Memory control hub 160 includes a control unit 240 coupled to a DRAM controller 250. DRAM controller 250 is coupled to memory chips 261A-261N. Control unit 240 includes an uplink control 241 and a downlink control 242. As noted above, memory bus 265 may be any type of memory interconnect. In the illustrated embodiment, memory control hub 160 is coupled to a memory link 110A in an upstream direction and a memory link 110B in a downstream direction. It is further noted that the frequency of operation of memory bus 265 is independent of the frequency of operation of memory links 110.

[0030] In the illustrated embodiment, uplink control unit 241 may be configured to receive and forward packets received from another memory module downstream. The receiving and forwarding of the upstream packets

creates an upstream transaction sequence. In addition, uplink control unit 241 may be configured to inject packets that originate within memory module 150 into the transaction stream.

[0031] In the illustrated embodiment, downlink control unit 242 may be configured to receive packets that originate at the host and if a memory module is connected downstream, to forward those packets to the downstream memory module. In addition, downlink control unit 242 may be configured to copy and decode the packets. In one embodiment, if the packets include an address that is within the range of addresses assigned to memory module 150 and the packet is a memory access request, downlink control unit 242 may pass the command associated with the packet to DRAM controller 250. In one embodiment, DRAM controller 250 is configured to initiate memory cycles to memory chips 261A-261N in response to memory commands from memory control hub 160. However, if the packet is not a memory request, but is instead a configuration packet, downlink control unit 242 may pass the configuration command associated with the packet to the core logic of control unit 240 (not shown) for processing. It is noted that in one embodiment, if the packet does not include an address that is within the range of addresses assigned to memory module 150, memory control hub 160 may drop or discard the packet if memory module 150 is the last memory module in the chain.

[0032] In one embodiment, memory control hub 160 is configured to receive a module present signal (not shown), which when activated by a downstream memory module, indicates to an upstream memory module that there is a downstream memory module present. In such an embodiment, if memory control hub 160 receives a transaction and no downstream memory module is determined to be present, memory control hub 160 may drop the transaction. In one specific implementation, when a given memory module 150 is inserted into a socket, the inserted memory module may apply a signal ground to the module present signal. Thus, an active module present signal is an active low signal.

[0033] It is noted that although the module present signal is active when a signal ground is applied in the embodiment described above, it is contemplated that in other embodiments, other voltage levels may be applied to the module present signal to indicate that a memory module is present.

[0034] Turning to FIG. 3, a block diagram of one embodiment of a downlink control unit is shown. In the embodiment illustrated in FIG. 3, downlink control unit 342 may be representative of the downlink control unit 242 illustrated in FIG. 2. Downlink control unit 342 is coupled to receive downstream transactions on downlink 312A and to transmit those downstream transactions on downlink 312B. In one embodiment, downlinks 312A and 312B may be representative of downlinks 212A and 212B of FIG. 2, respectively. It is noted that, as described above, downlink 312A and 312B include multiple bits. For example, in one implementation, downlinks 312A and 312B may each be 18-bit downlinks. In addition, downlink control unit 342 is configured to receive busy signal 371 from an upstream memory module or host 100 and to transmit busy signal 372 to a downstream memory module.

[0035] In the illustrated embodiment, downlink control unit 342 includes a phase alignment unit 310 which receives the incoming transactions. Phase alignment unit 310 is coupled to a transmit unit 315 and to a data recovery unit 320. Data recovery unit 320 is coupled to a synchronization first in first out buffer (FIFO) 325, which is coupled to address decode logic 330. In one embodiment, sync FIFO 325 is coupled to a DRAM controller such as DRAM controller 250 and to the core logic within control unit 240 of FIG. 2. It is noted that in the embodiment illustrated in FIG. 3, phase alignment unit 310, transmit unit 315 and data recovery unit 320 may operate on each bit of the downlink independently in a bit-wise manner. It is contemplated that in other embodiments, phase alignment

unit 310, transmit unit 315 and data recovery unit 320 may operate on all the bits of the downlink at substantially the same time.

[0036] To provide the necessary memory bandwidth, it may be important for memory control hub 160 to efficiently forward received transactions to downstream memory modules. Accordingly, transactions received by downlink control unit 342 are blindly forwarded downstream. In one embodiment, each received bit is phase aligned by phase alignment unit 310 and sent to transmit unit 315, without decoding, for transmission on downlink 312B. In addition, each received bit is sent to data recovery unit 320 for processing.

[0037] In one embodiment, phase alignment unit 310 is configured to, for each bit, dynamically adjust the phase of the local sampling clock to allow downlink control unit 342 to sample in the middle of the received data eye. For example, in one embodiment, a reference clock is provided from the host to a phase locked loop (PLL) (not shown) within each memory control hub 160. The PLL generates the local sampling clock as well as the transmit clock within downlink control unit 342. Phase alignment unit 310 is configured to track the average phase of the incoming data stream. Phase alignment unit 310 may also be configured to compensate for static skew caused by processing variations and to track low frequency variations of data phase caused by voltage and temperature variations and to compensate for any low frequency phase jitter that may be on the reference clock and not tracked by the hub's PLL. Phase alignment unit 310 filters the high frequency jitter caused by inter-symbol interference, cross-talk or reflective noise and high frequency phase noise that may be generated by the transmitter.

[0038] To facilitate phase alignment of the local sampling clock, it may be necessary to provide a sufficient number of transitions (or transmission density) on each bit of downlink 312A. In one embodiment, the transition density is provided by scrambling the data transmitted on each bit and then de-scrambling the received data upon receipt. In one embodiment, the data is scrambled by Exclusive-ORing (XOR) the data with a pseudo random binary sequence (PRBS). A linear feedback shift register (LFSR) may generate the PRBS using a given seed polynomial. In another embodiment, the transition density may be provided by sending synchronization packets that have predetermined data payloads. In such an embodiment, the predetermined data payloads may include a random or pseudo-random pattern that has a sufficient number of transitions.

[0039] Regardless of the way in which the transition density is obtained, data recovery unit 320 is configured to recover the data bits. In embodiments that use scrambling, data recovery unit 320 may be configured to de-scramble the received data using the same PRBS as that used to scramble the data. Conversely, in embodiments that use synchronization packets to obtain the transition density, data recovery unit 320 may simply drop or discard the synchronization packet data once phase alignment unit 310 has finished aligning the local sampling clock.

[0040] In the illustrated embodiment, sync FIFO 325 is configured to store the incoming data bits for use by the core logic of control unit 240. For example, since the transactions are sent in packetized fashion, the received bits may be stored in sync FIFO 325 and re-formatted for decoding. The received packets are then decoded by address decode logic 330. As noted above, each memory module 150 may be assigned a range of memory addresses. In addition, each memory control hub 160 may be assigned a hub address for use in configuration transactions. If a transaction address is decoded and a match is indicated, the command information may be extracted and processed. In one embodiment, if the transaction is a memory access request having an address that matches a memory address associated with memory control hub 160, the decoded command may be sent to DRAM controller 250. If the transaction is a configuration transaction, the decoded command may be sent to the core logic

of control unit 240. In addition, if address decode logic 330 does not detect a match, the packet may be discarded or dropped.

[0041] In the illustrated embodiment, phase alignment unit 335 is coupled to receive the incoming busy signal 371. Phase alignment unit 335 is also coupled to data recovery unit 350, which is in turn coupled to busy pulse injection unit 355. Busy pulse injection unit 355 is coupled to transmit unit 360.

[0042] In the illustrated embodiment, phase alignment unit 335 and data recovery unit 350 operate similar to phase alignment unit 310 and data recovery unit 320, respectively. However, in one embodiment, busy signal 371 is a single bit differential signal and not a multi-bit signal. Busy signal 371 and busy signal 372 are used to ensure that hubs near the bottom of the serial chain of memory modules will leave enough idle time for hubs near the top of the serial chain to inject packets. In one embodiment, busy signal 371 includes an indication of the number of packets that will be injected into the upstream transaction sequence by all memory modules that are upstream. Accordingly, busy signal 372 includes an indication of the total number of packets that will be injected into the upstream transaction sequence by all memory modules that are upstream and the number of packets that will be injected locally by memory control hub 160.

[0043] In one embodiment, busy pulse injection unit 355 is configured to receive data corresponding to the number of packets that will be injected into the upstream transaction sequence by all memory modules that are upstream. In addition, busy pulse injection unit 355 is configured to receive an indication from the core logic of control unit 240 corresponding to the number of packets that will be injected into the upstream transaction sequence locally by memory control hub 160. Accordingly, busy pulse injection unit 355 is configured to create busy signal 372. In one embodiment, busy signals 371 and 372 may include a pulse for each injected packet. Thus, a signal such as busy signal 372 that is indicative of five packets being injected would include five pulses, for example. However it is contemplated that in other embodiments, any number of pulses may be included for each injected packet. As will be described further below, packets may be sent as short packets or long packets. Accordingly, in one embodiment, for an injected long packet, busy pulse injection unit 355 may include two pulses for each long packet.

[0044] In addition, busy pulse injection unit 355 is configured to provide an indication corresponding to the number of busy pulses received from upstream to uplink control unit 241 of FIG. 2. In one embodiment, if a memory module is the last module in the serially connected chain, uplink control unit 241 may cause one or more NOP packets to be injected into the upstream transaction sequence dependent upon the number of busy pulses received on busy signal 371. For example, in one implementation, for every two busy pulses received, uplink control unit 241 may cause one NOP packet to be injected into the upstream transaction sequence. However, other embodiments are contemplated in which other ratios of NOP packets to received busy pulses may be used.

[0045] In the illustrated embodiment, transmit unit 360 is configured to receive the busy signal from busy pulse injection unit 355. In one embodiment, transmit unit 360 is further configured to scramble the data, as described above, using an LFSR prior to transmitting busy signal 372. Alternatively, transmit unit 360 may inject synchronization packets as described above.

[0046] Referring to FIG. 4, a block diagram of one embodiment of an uplink control unit is shown. In the embodiment illustrated in FIG. 4, uplink control unit 441 may be representative of the uplink control unit 241 illustrated in FIG. 2. Uplink control unit 441 is coupled to receive upstream transactions on uplink 411B and to

transmit upstream transactions on uplink 411A. In addition, uplink control unit 241 is configured to inject packets into the upstream transaction sequence received on uplink 411B. In one embodiment, uplinks 411A and 411B may be representative of uplinks 211A and 211B of FIG. 2, respectively. It is noted that, as described above, uplink 411A and 411B include multiple bits. For example, in one implementation, uplinks 411A and 411B may each be 16-bit uplinks.

[0047] In the illustrated embodiment, uplink control unit 441 includes a phase alignment unit 410 which receives the incoming upstream transactions. Phase alignment unit 410 is coupled to a data recovery unit 415.

[0048] To provide the necessary memory bandwidth, it may be important for memory control hub 160 to efficiently forward received upstream transactions to upstream memory modules or to the host. Similar to downstream control unit 342, transactions received by uplink control unit 441 are forwarded upstream. However, in contrast to downstream control unit 342, uplink control unit 441 may inject locally generated packets into the upstream transaction sequence.

[0049] In the illustrated embodiment, phase alignment unit 410 operates similar to phase alignment unit 310. Accordingly, phase alignment unit 410 is configured to, for each bit, dynamically adjust the phase of the local sampling clock to allow uplink control unit 441 to sample in the middle of the received data eye as described above. Similarly, data recovery unit 415 is configured to recover the data bits as described above in conjunction with the description of data recovery unit 320.

[0050] In the illustrated embodiment, data recovery unit 415 is coupled to multiplexer 430, which is coupled to transmit unit 435. If uplink control unit 441 has no packets to inject into the upstream transaction sequence, uplink control unit 441 is configured to forward received transactions on uplink 411A. In one embodiment, packet boundary track unit 450 may provide control signals to multiplexer 430 that may allow the received transactions to be scrambled and transmitted by transmit unit 435. In one embodiment, transmit unit 435 may scramble the data to be transmitted using an LFSR as described above, while in other embodiments, transmit unit 435 may inject synchronization packets as described above.

[0051] However, in one embodiment, as packets to be injected are received and stored within injection FIFO 425 from hub core logic, the received data from data recovery unit 415 is stored within holding FIFO 420 for later transmission. When the transmission of the current forwarded packet is complete, data stored within injection FIFO 425 is provided to an input of multiplexer 430. In addition, packet boundary track unit 450 tracks the packet boundaries of packets received from uplink 411B. Packet boundary track unit 450 ensures that the injected packets do not collide with the forwarded packets by providing control signals to multiplexer 430 at the correct time; thereby allowing the waiting packet to be injected to be scrambled and transmitted by transmit unit 435. Assuming that there are packets to be injected stored within injection FIFO 425 and packets to be forwarded stored within holding FIFO 420, packet boundary track unit 450 may selectively control multiplexer 430 to pass either type of packet to transmit unit 435 depending upon a fairness algorithm. The fairness algorithm may be any algorithm configured to prevent starvation of either the injected packets or the forwarded packets for any given memory module. It is noted that NOP packets received on uplink 411B may be discarded instead of being stored within holding FIFO 420 or being forwarded.

[0052] It is noted that in an alternative embodiment, to facilitate improved latency for forwarded transaction traffic, phase alignment unit 410 may be coupled to holding FIFO 420 and to multiplexer 430, as indicated by the dashed lines. Such an embodiment may provide improved latency when no injected traffic is present.

[0053] It is further noted that in one embodiment, prior to communication between host 100 and each memory control hub 160 of each memory module 150, an initialization sequence may be performed to synchronize any scramble/de-scramble logic, sync FIFOs, and phase alignment logic that may be present within host 100 and within each memory control hub 160. For example, in one embodiment, during reset, the initialization sequence may include host 100 sending one or more synchronization patterns downstream. The synchronization patterns may be forwarded back upstream by each memory control hub 160 to host 100. The synchronization patterns may be used to allow receivers including scramble/de-scramble logic to lock synchronize.

The Memory Interconnect

[0054] Referring collectively to FIG. 1 and FIG. 2, the memory interconnect includes one or more high-speed point-to-point memory links such as memory links 110A-110C each including an uplink such as uplink 211A and a downlink such as downlink 212A, for example. As noted above, in one embodiment downlinks may be 18-bit links while uplinks may be 16-bit links. As such, an 18-bit downlink may include 16 control, address and data (CAD) signals, a busy signal and a Control (CTL) signal. A given uplink may include 16 control, address and data (CAD) signals. It is contemplated however, that in an alternative embodiment, an uplink such as uplink 211A may also include a CTL signal.

[0055] In addition to the high-speed links, other signals may be provided to each memory module 150. For example, in one embodiment, a reset signal, a power OK signal and a reference clock may be provided to each memory module 150 from host 100. Further, other signals may be provided between each memory module. For example, as described above, a next memory module present signal may be provided between memory modules.

[0056] Generally speaking, the types of transactions conveyed on memory links 110 may be categorized into configuration and control transactions and memory transactions. In one embodiment, configuration and control transactions may be used to configure memory control hub 160. For example, configuration and control transactions may be used to access configuration registers, assign a memory address range to a memory module or to assign a hub address to a memory control hub. Memory transactions may be used to access the memory locations within memory chips 261A-261N.

[0057] Accordingly, in one embodiment, there are two types of addressing supported: hub addressing and memory addressing. Using hub addressing, eight hub bits identify the specific memory control hub being accessed. In one embodiment, a hub address of FFh may be indicative of a broadcast to all memory control hubs. Using memory addressing, each hub decodes the upper portion of the address bits to determine which hub should accept the request and the lower portion to determine the memory location to be accessed. In one embodiment, there are 40 address bits, although it is contemplated that other numbers of address bits may be used as desired.

[0058] In one embodiment, each of the memory links is configured to convey the transactions using one or more packets. The packets include control and configuration packets and memory access packets, each of which may include a data payload depending on the type of command the packet carries. As such, the sets of wires that make up memory links 110 may be used to convey control, address and data.

[0059] The packets may be generally characterized by the following: Each packet includes a number of bit positions which convey a single bit of information. Each packet is divided into several bit times and during a given bit time, all of the bit positions of the packet are sampled. As such, the control information and data share the same wires of a given link (e.g., CAD wires). As will be described in greater detail below, in one embodiment, packets are multiples of bit pairs and the first bit-time of every packet is sampled at an even bit-time. Packets begin with a control header that may be either one or two bit-pairs in length. In one embodiment, the first five bits of the control header is the command code. Table 1 below illustrates the various types of packets and their associated command codes. It is noted however, that the actual codes shown in column one are for illustrative purposes and that other codes may be used for each given command.

Table 1. Packet types and command codes

Code	Header Length (bit-times)	Command	Description	Direction	Normal Response	Address Type
00h	-	NOP	Null Operation/Idle State	Both	-	-
04h	2	AddrSet	Address Set	Down	AddrAck	Hub
05h	2	AddrAck	Address Acknowledge	Up	-	-
06h	2	Ack	Acknowledge	Up	-	-
07h	2	Nak	Not Acknowledge/Error	Up	-	-
08h	2	SRdResp	Short Read Response	Up	-	-
09h	2	LRdResp	Long Read Response	Up	-	-
0Ah	2	ConfigRd	Configuration Read	Down	RdResp	Hub
0Ch	2	ConfigWr	Configuration Write	Down	Ack	Hub
0Eh	2	DIMMCtl	DIMM Control	Down	Ack	Hub
10h	4	SMemRd	Short Memory Read	Down	RdResp/Ack	Memory
11h	4	LMemRd	Long Memory Read	Down	RdResp	Memory
12h	4	BlkMemWr	Block Memory Write	Down	Ack	Memory
13h	4	SbytMemWr	Short Byte Memory Write	Down	Ack	Memory
14h	4	LbytMemWr	Long Byte Memory Write	Down	Ack	Memory

[0060] Further, in one embodiment, packets (except NOP packets) are transmitted with an error detecting code (EDC). It is noted that in one embodiment, the EDC is a 32-bit cyclic redundancy code (CRC), although other embodiments may employ other EDC's as desired. Additionally, addresses are sent most significant bit-time first to speed decode within memory control hub 160 while data is sent least significant byte first. It is noted however, that other embodiments are contemplated in which the addresses may be sent least significant bit-time first and data may be sent most significant byte first. Packets may carry a payload of byte enables and/or data. Packets with no payload are referred to as header-only packets. In one embodiment, the size of the data short reads may be up to one half of a programmed cache line size. In addition, the size of the data for long reads and block writes may be up to

the programmed cache line size. Further, the size of the data for byte writes may be a maximum of 64 bytes regardless of the cache line size setting.

[0061] In addition to the control header and command code information included within a packet, the CTL signal may be used to convey information about each packet. As illustrated in Table 2 below, some exemplary CTL encodings are shown.

Table 2. CTL encodings for downstream use

Even	Odd	Content of CAD
0	0	Data or Byte Enable Payload
1	1	Control Header
0	1	CRC for a Packet with Payload
1	0	CRC for a Header-Only Packet

[0062] Different values of CTL for the header and payload portions of a packet may provide enough information to allow header-only packets to be inserted within the payload of another packet. This may be useful for reducing the latency of read commands by allowing them to issue while a write packet is still being sent on the link.

Table 3 illustrates an exemplary packet including a payload in tabular format. The packet in table 3 also shows that a header-only packet is inserted in the payload during bit times 4-7.

Table 3. Packet with payload and header-only packet inserted within payload

Bit-time	CTL	CAD
0	1	Header1 bits [15: 0]
1	1	Header1 bits [31:16]
2	0	Data bits [15: 0]
3	0	Data bits [31:16]
4	1	Header2 bits [15: 0]
5	1	Header2 bits [31:16]
6	1	CRC2 bits [15: 0]
7	0	CRC2 bits [31:16]
8	0	Data bits [47:32]
9	0	Data bits [64:48]
10	0	CRC1 bits [15: 0]
11	1	CRC1 bits [31:16]

[0063] FIG. 5A through FIG. 5D illustrate exemplary packets that may be conveyed on memory links 110A through 110C of FIG. 1. Turning now to FIG. 5A, a diagram of one embodiment of a configuration read packet is shown. In the illustrated embodiment, configuration read packet 510 is 16 bits wide and includes four bit times or two bit-pairs.

[0064] During bit time zero, the five-bit command code (e.g., 0Ah) is conveyed in bit positions 0-4. Bit positions 5-7 are reserved. An eight-bit tag is conveyed in bit positions 8-15. In one embodiment, the tag value may be used by host 100, for example, to match an upstream response packet with an outstanding downstream request. If unused, the tag field may be set to a value of 00h. During bit time one, the eight-bit hub address is conveyed in bit positions 0-7. In addition, an eight-bit configuration register number is conveyed in bit positions 8-15. During bit times two and three, bits 0-15 and 16-31, respectively, of a CRC are conveyed in bit positions 0-15.

[0065] Referring to FIG. 5B, a diagram of one embodiment of a configuration write packet is shown. In the illustrated embodiment, configuration write packet 515 is 16 bits wide and includes six bit times or three bit-pairs.

[0066] During bit time zero, the five-bit command code (e.g., 0Ch) is conveyed in bit positions 0-4. Bit positions 5-7 are reserved. An eight-bit tag is conveyed in bit positions 8-15. During bit time one, the eight-bit hub address is conveyed in bit positions 0-7. In addition, the eight-bit configuration register number is conveyed in bit positions 8-15. During bit times two and three, bits 0-15 and 16-31 of a data payload are conveyed in bit positions 0-15. During bit times four and five, bits 0-15 and 16-31, respectively, of a CRC are conveyed in bit positions 0-15.

[0067] Turning to FIG. 5C, a diagram of one embodiment of a memory read packet is shown. In the illustrated embodiment, memory read packet 520 is 16 bits wide and includes six bit times or three bit-pairs. During bit time zero, the five-bit command code (e.g., 10h or 11h) is conveyed in bit positions 0-4. Bit positions 5-7 are reserved. An eight-bit tag is conveyed in bit positions 8-15.

[0068] During bit time one, the length of the data that should be returned conveyed in bit positions 0-5. In one embodiment, a value of 00h indicates no data, a value of 01h indicates two bit-pairs of data, a value of 02h indicates four bit-pairs of data, and so on. A zero length read results in an acknowledge packet (Ack) being returned to the requestor. In one embodiment, a read of a half cache line or less may result in a short RdResp and a read of more than a half cache line may result in either a single long RdResp or two short RdResp. The cache line size may be programmed by software into the configuration registers of host 100 and each memory control hub 160. Bits 6-7 are reserved. Address bits 39-32 of the requested location in DRAM are conveyed in bit positions 8-15.

[0069] During bit time two, the address bits 31-16 of the requested location in DRAM are conveyed in bit positions 0-15 and during bit time 3, the address bits 3-15 of the requested location in DRAM are conveyed in bit positions 3-15. Also during bit time 3, the packet priority is conveyed in bit positions 0-1. In one embodiment, the priority may be indicative of the priority of the packet relative to other requests. For example, one priority may be to delay all requests with lower priority even if they are already in progress and to execute this request ahead of them. Bit position 2 is reserved. During bit times four and five, bits 0-15 and 16-31, respectively, of a CRC are conveyed in bit positions 0-15.

[0070] Referring to FIG. 5D, a diagram of one embodiment of a block memory write packet is shown. In the illustrated embodiment, block memory write packet 525 is 16 bits wide and includes eight bit times or four bit-pairs. During bit time zero, the five-bit command code (e.g., 12h) is conveyed in bit positions 0-4. Bit positions 5-7 are reserved. An eight-bit tag is conveyed in bit positions 8-15.

[0071] During bit time one, the length of the data being conveyed in the data payload is conveyed in bit positions 0-5. In one embodiment, a value of 00h indicates no data, a value of 01h indicates two bit-pairs of data, a value of 02h indicates four bit-pairs of data, and so on. Bits 6-7 are reserved. Address bits 39-32 of the location in DRAM being written are conveyed in bit positions 8-15.

5 **[0072]** During bit time two, the address bits 31-16 of the location in DRAM being written are conveyed in bit positions 0-15 and during bit time 3, the address bits 3-15 of the location in DRAM being written are conveyed in bit positions 3-15. Also during bit time 3, the packet priority is conveyed in bit positions 0-1. Bit position 2 is reserved.

10 **[0073]** During bit times four and five, bits 0-15 and 16-31 of a first bit pair of the data payload are conveyed in bit positions 0-15. If more data is being written, subsequent bit pairs may convey bits 0-15 and 16-31 of subsequent data payload. During bit times 4+2N and 5+2N, bits 0-15 and 16-31, respectively, of a CRC are conveyed in bit positions 0-15.

15 **[0074]** It is noted that although only four types of packets were shown, other types of packets, which may correspond to the command codes listed in table 3 are contemplated. It is further noted that although the various fields of the exemplary packets are shown having a particular number of bits, it is contemplated that in other embodiments, the various fields of the packet may include other numbers of bits as desired.

20 **[0075]** FIG. 6 is a block diagram of one embodiment of a computer system. Computer system 600 includes process nodes 612A-612D each interconnected by coherent packet interface links 615A-D. Each link of coherent packet interface 615 may form a high-speed point-to-point link. Process nodes 612A-D may each include one or more processors. Computer system 600 also includes an I/O node 620 which is coupled to process node 612A via a non-coherent packet interface 650A. I/O node 620 may be connected to another I/O node (not shown) in a chain topology for example, by non-coherent packet interface 650B. Process nodes 612A is illustrated as a host node and may include a host bridge for communicating with I/O node 620 via NC packet interface 650A. Process nodes 612B-D may also include host bridges for communication with other I/O nodes (not shown). The non-coherent packet interface links formed by NC packet interface 650A-B may also be referred to as point-to-point links. I/O node 620 is connected to a pair of peripheral buses 625A-B.

25 **[0076]** FIG. 6 further illustrates respective system memories (e.g., 617A and 617B) coupled to process nodes 612A and 612B. In the illustrated embodiment, process node 612A and 612B are each illustrative of a host as shown in FIG. 1, and each system memory 617 may be implemented in the configuration described in conjunction with the description of FIG. 1 and FIG. 2 above. Further, the interconnects between each of process nodes 612A and 612B and their respective system memories 617 may be reflective of the memory interconnect including memory link 110C described above in FIG. 1 and FIG. 2. It is noted that in other embodiments, other numbers of process nodes may be used. Further, it is contemplated that each of process nodes 612C and 612D may be similarly connected to a respective system memory such as system memory 617, for example.

35 **[0077]** In the illustrated embodiment, each link of coherent packet interface 615 is implemented as sets of unidirectional lines (e.g. lines 615A are used to transmit packets from processing node 612A to processing node 612B and lines 615B are used to transmit packets from processing node 612B to processing node 612C). Other sets of lines 615C-D are used to transmit packets between other processing nodes as illustrated in FIG. 1. The coherent packet interface 615 may be operated in a cache coherent fashion for communication between processing nodes

("the coherent link"). Further, non-coherent packet interface 650 may be operated in a non-coherent fashion for communication between I/O nodes and between I/O nodes and a host bridge such as the host bridge of process node 612A ("the non-coherent link"). The interconnection of two or more nodes via coherent links may be referred to as a "coherent fabric". Similarly, the interconnection of two or more nodes via non-coherent links may be referred to as a "non-coherent fabric". It is noted that a packet to be transmitted from one processing node to another may pass through one or more intermediate nodes. For example, a packet transmitted by processing node 612A to processing node 612C may pass through either processing node 612B or processing node 612D as shown in FIG. 6. Any suitable routing algorithm may be used. Other embodiments of computer system 600 may include more or fewer processing nodes than the embodiment shown in FIG. 6.

[0078] One example of a packet interface such as non-coherent packet interface 650 may be compatible with HyperTransport™ technology. Peripheral buses 625A and 625B are illustrative of a common peripheral bus such as a peripheral component interconnect (PCI) bus. It is understood, however, that other types of buses may be used.

[0079] It is further noted that other computer system configurations are possible and contemplated. For example, it is contemplated that the system memory configuration described above in FIG. 1 through FIG. 5 may be used in conjunction with a computer system employing a processor chipset that includes a Northbridge. In such an embodiment, a memory controller within the Northbridge may serve as the host.

[0080] Although the embodiments above have been described in considerable detail, numerous variations and modifications will become apparent to those skilled in the art once the above disclosure is fully appreciated. It is intended that the following claims be interpreted to embrace all such variations and modifications.

Industrial Applicability

This invention may generally be applicable to computer system memory.

WHAT IS CLAIMED IS:

1. A system (50) comprising:
a host (100); and
5 a plurality of memory modules (150A, 150B) coupled serially in a chain to said host.
2. The system as recited in claim 1, wherein each of said plurality of memory modules includes a memory control hub (160A, 160B) coupled to control access to a plurality of memory chips (261A-N).
- 10 3. The system as recited in claim 2, wherein said plurality of memory modules is coupled serially in a chain to said host via a plurality of memory links (110A, B, C), wherein each memory link includes an uplink (211A) for conveying transactions toward said host and a downlink (212A) for conveying transactions originating at said host to a next memory module in said chain.
- 15 4. The system as recited in claim 3, wherein said uplink and said downlink are each a uni-directional link including a plurality of signals configured to convey transactions using packets that include control and configuration packets and memory access packets, wherein at least a portion of packets include control, address and data information, and wherein said control, address and data information share the same wires of a given link.
- 20 5. The system as recited in claim 3, wherein said memory control hub is configured to receive a transaction on a first downlink (212A) of a first memory link and convey said transaction on a second downlink (212B) of a second memory link independent of decoding said transaction.
- 25 6. The system as recited in claim 5, wherein said memory control hub includes a DRAM controller (250) configured to access said plurality of memory chips in response to decoding said transaction into a memory command having a memory address that matches a memory address associated with said memory control hub.
7. The system as recited in claim 3, wherein said memory control hub is configured to receive a transaction on a first uplink of a first memory link and to convey said transaction on a second uplink of a second memory link, and
30 wherein said memory control hub includes a control unit configured to selectively inject a local transaction into a sequence of transactions being conveyed on said second uplink.
8. The system as recited in claim 1, wherein said host includes a memory controller (105) configured to issue memory request transactions without knowledge of a memory size associated with each of said memory modules or
35 an address range associated with any of said memory modules.
9. The system as recited in claim 8, wherein said memory controller is further configured to issue a memory read request transaction prior to receiving a response to a previous memory read request transaction.
- 40 10. A method comprising:

connecting a host (100) to a first memory module (150A) via a first memory link (110A); and
connecting said first memory module to a second memory module (150B) via a second memory link
(110B);

wherein said first memory module and said second memory module are connected to said host serially via
said a first memory link and said second memory link; and

wherein said first memory module is configured to receive a transaction on said first memory link and to
convey said transaction on said second memory link independent of decoding said transaction.

10

50

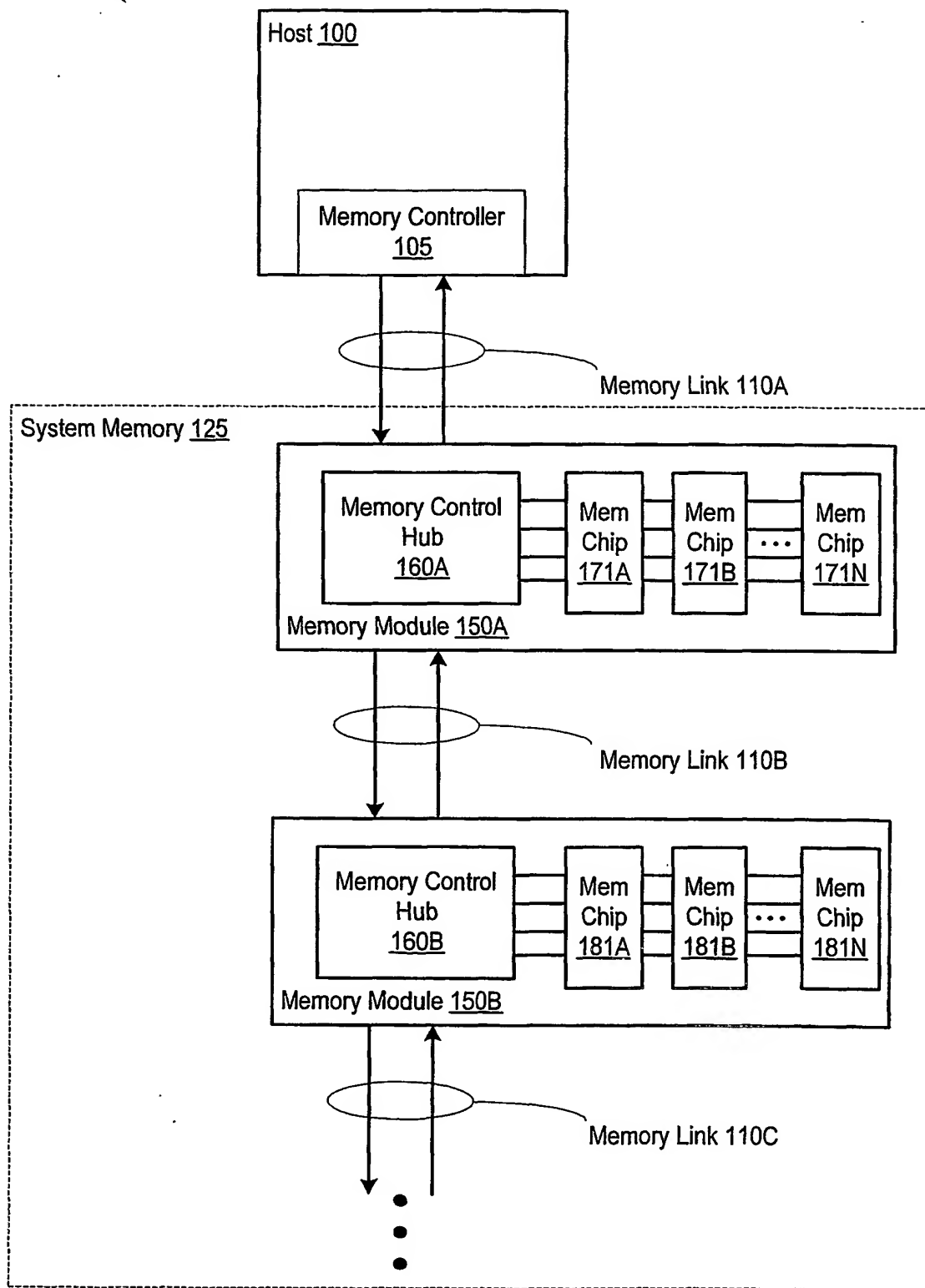


FIG. 1

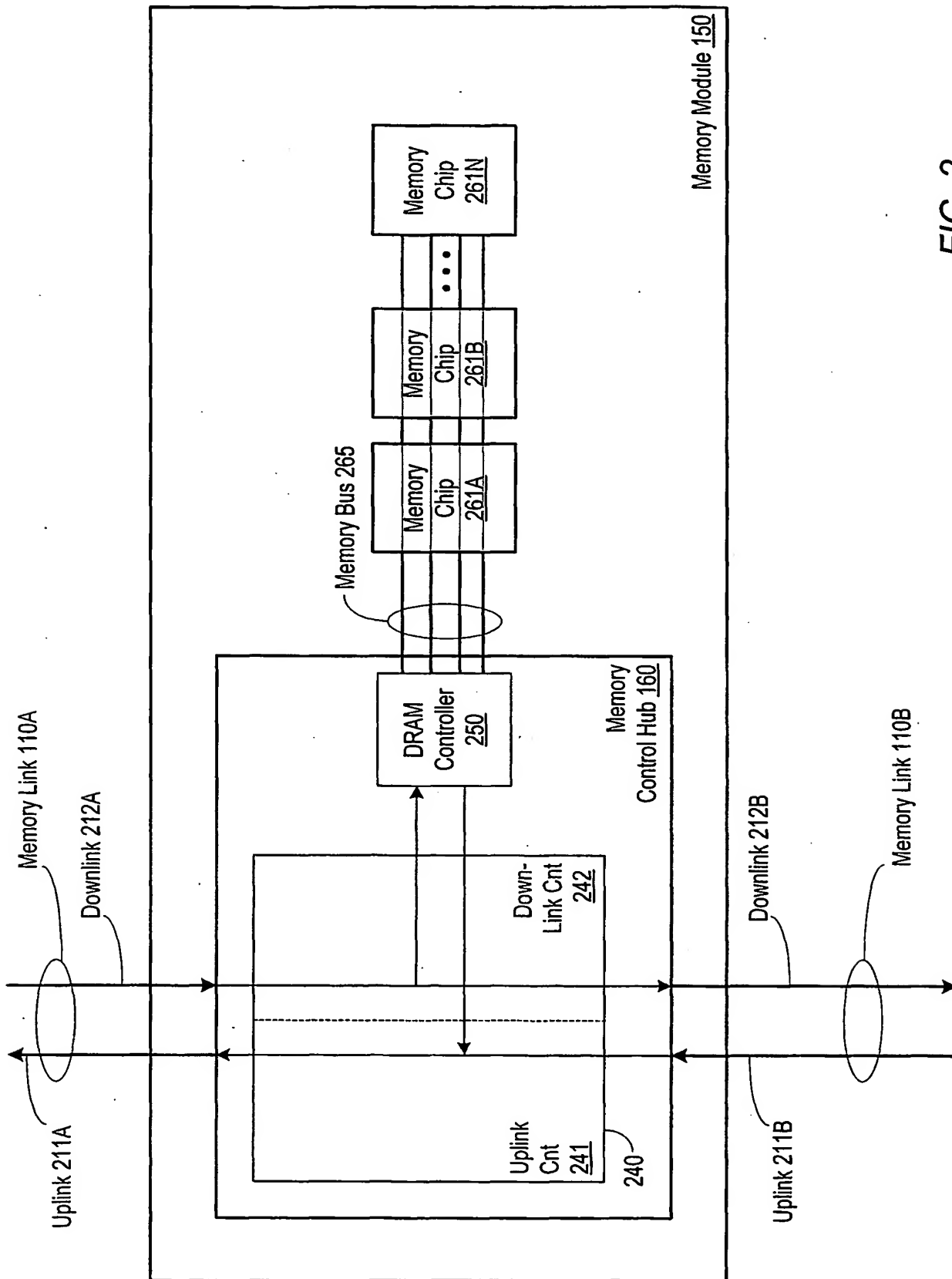


FIG. 2

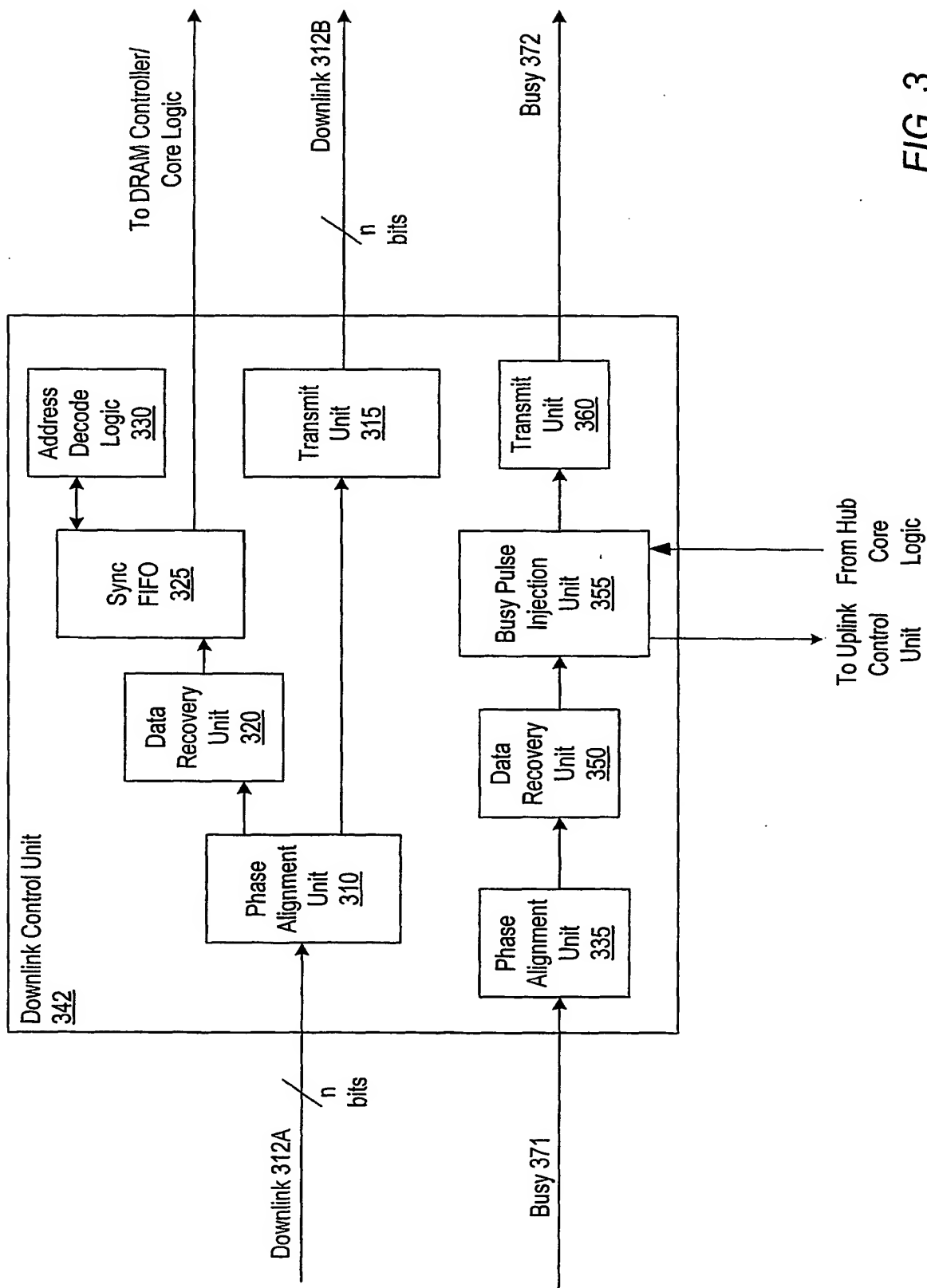


FIG. 3

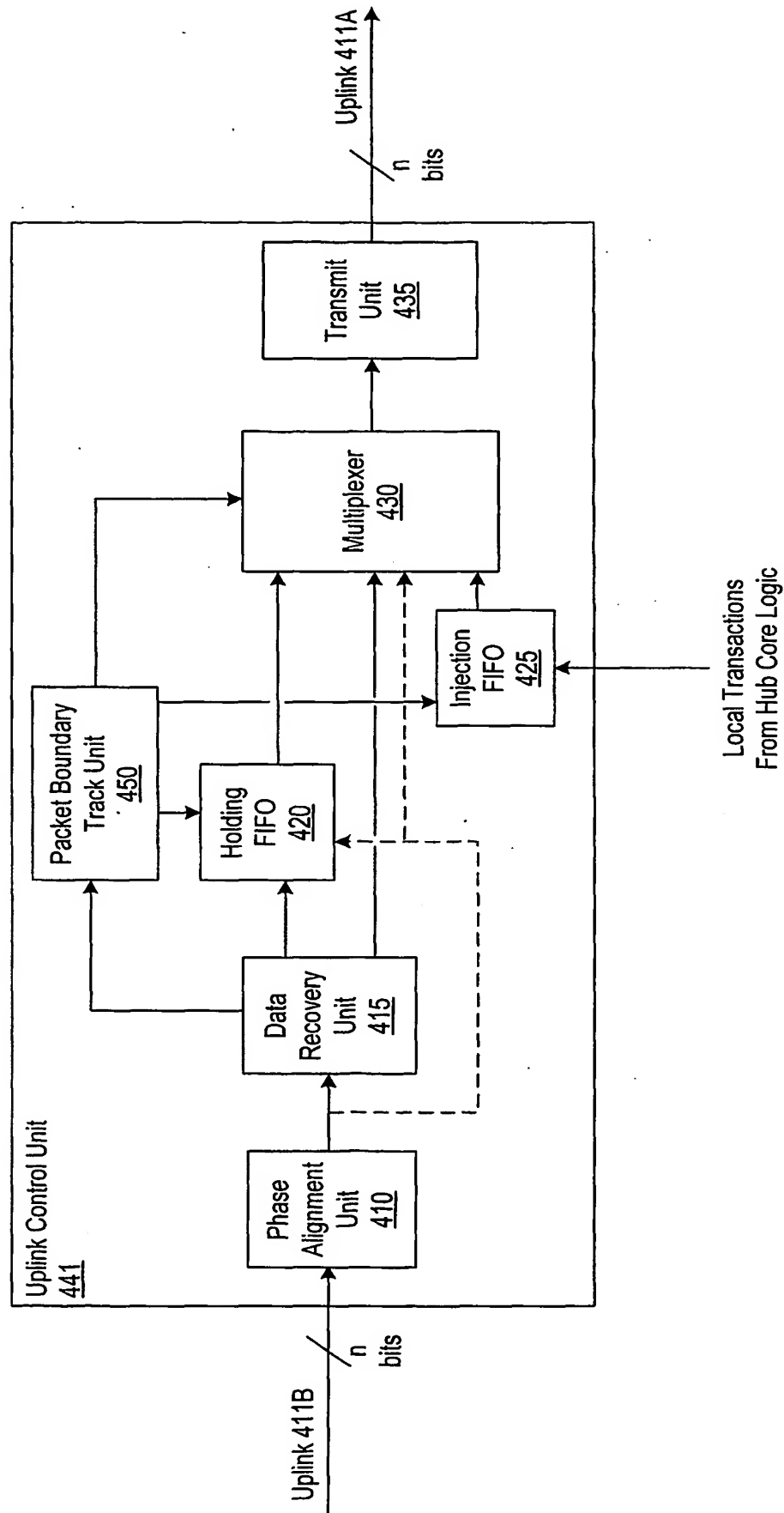


FIG. 4

Configuration Read Packet

510

BIT TIME	15	8	7	5	4	0
0	Tag [7:0]		RSV		Command Code (0Ah)	
1	Register [7:0]				Hub [7:0]	
2		CRC [15:0]				
3		CRC [31:16]				

FIG. 5A

Configuration Write Packet

515

BIT TIME	15	8	7	5	4	0
0	Tag [7:0]		RSV		Command Code (0Ch)	
1	Register [7:0]				Hub [7:0]	
2		Data [15:0]				
3		Data [31:16]				
4		CRC [15:0]				
5		CRC [31:16]				

FIG. 5B

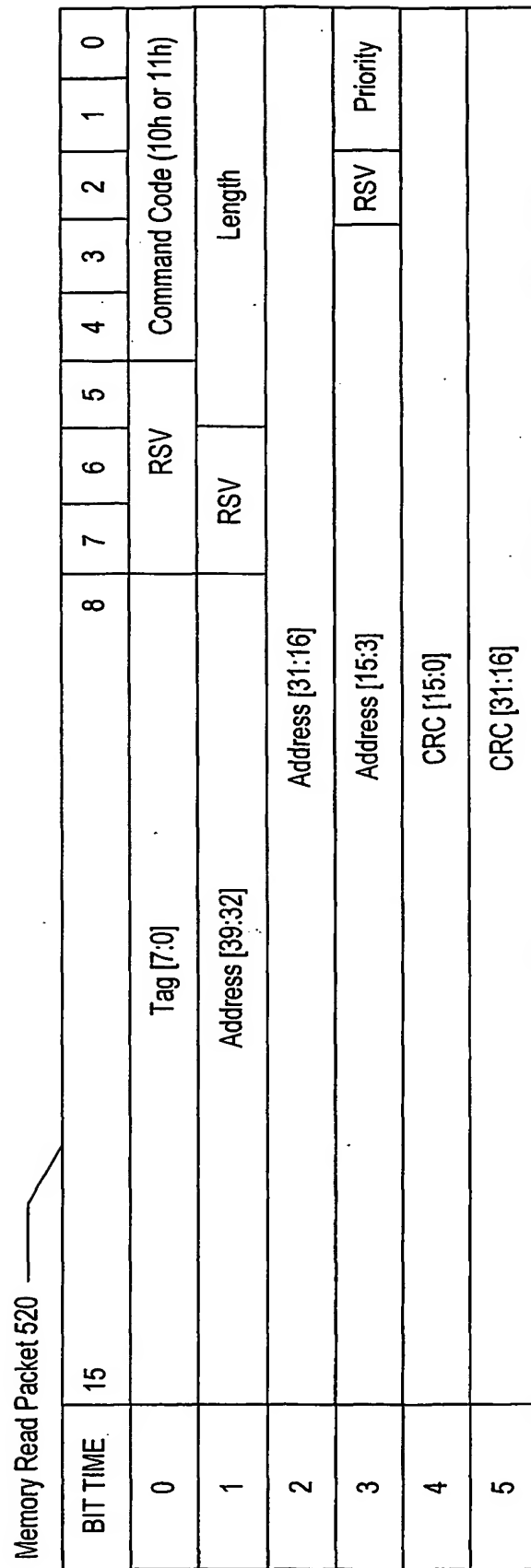



FIG. 5C

Memory Write Packet 525

BIT TIME	15	8	7	6	5	4	3	2	1	0
0		Tag [7:0]		RSV						Command Code (12h)
1		Address [39:32]		RSV						Length
2										
3										
4										
5										
...										
4+2N										
5+2N										

FIG. 5D

600 

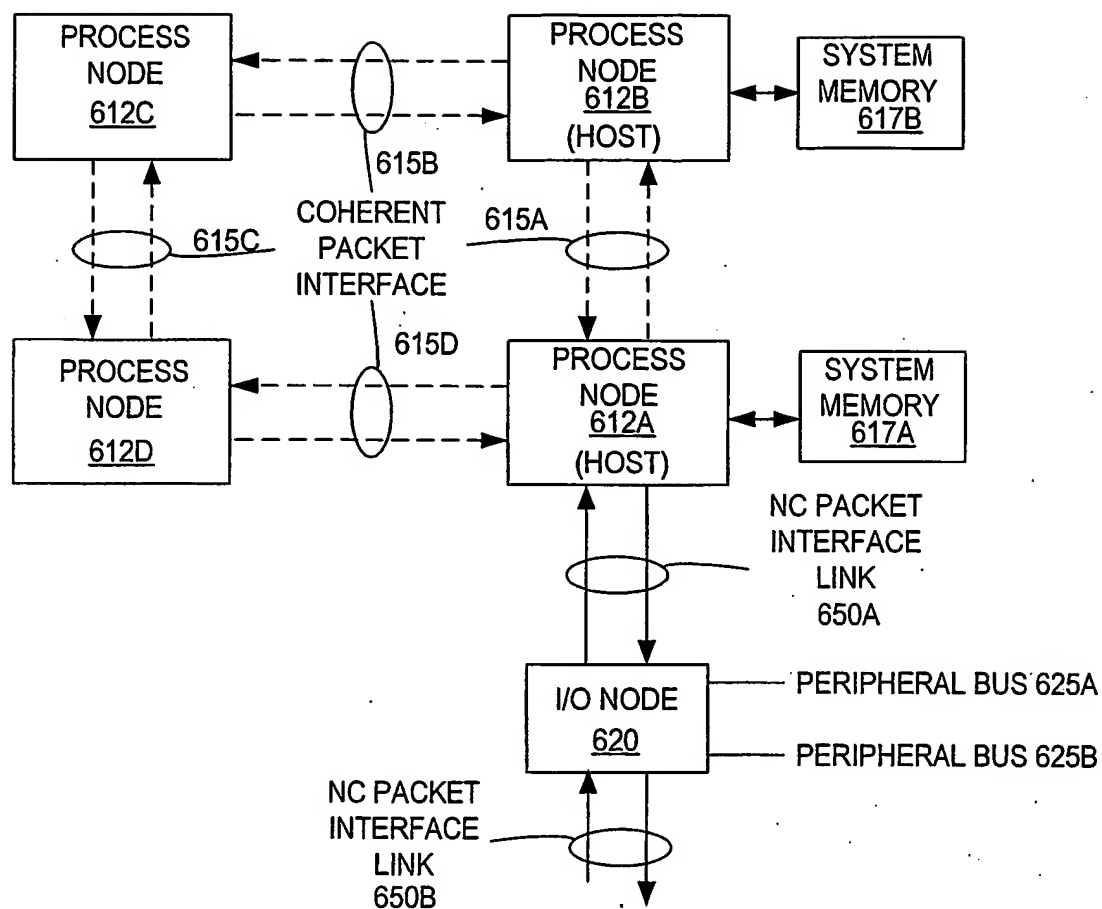


FIG. 6

THIS PAGE BLANK (USPTO)